

DEVICE, PROGRAM, AND METHOD FOR MAKING PROGRAM DIFFICULT TO BE READ

Publication number: JP2005049925

Publication date: 2005-02-24

Inventor: KADOTA AKIHITO; SATO HIROTSUGU; KANZAKI YUICHIRO

Applicant: NARA INST OF SCIENCE & TECHNOL

Classification:

- international: G06F12/14; G06F1/00; G06F12/00; G06F21/22;
G06F12/14; G06F1/00; G06F12/00; G06F21/22; (IPC1-7): G06F12/00; G06F1/00; G06F12/14

- european:

Application number: JP20030202795 20030729

Priority number(s): JP20030202795 20030729

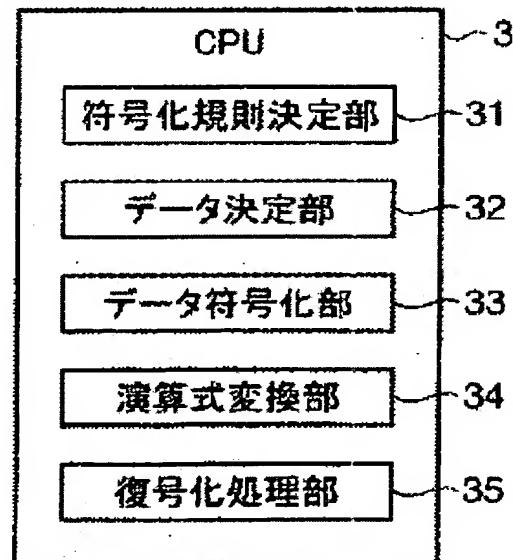
[Report a data error here](#)

Abstract of JP2005049925

PROBLEM TO BE SOLVED: To make a program difficult to be read by a method for making an object difficult to be read which makes secret data or algorithm difficult to be read, can also make the acquisition of secret data of a program by prying a stack memory difficult and further can be applicable to a general program.

SOLUTION: A device for making a program difficult to be read comprises; a data encoding section 33 for encoding data in a program using an encoding rule $C(a, b)=ax+b$; an arithmetic expression conversion section 34 for converting an arithmetic expression into a coded arithmetic expression by applying an operator conversion rule $R(a, b)-R10(a, b)$ to each operator of an arithmetic expression including coded data encoded by the data encoding section 33; and a decoding processing section 35 for changing a coded arithmetic expression into a decoded arithmetic expression using a decoding rule $D(a, b)$.

COPYRIGHT: (C)2005,JPO&NCIPI



Data supplied from the esp@cenet database - Worldwide

THIS PAGE BLANK (USPTO)

(19) 日本国特許庁(JP)

(12) 公開特許公報(A)

(11) 特許出願公開番号

特開2005-49925

(P2005-49925A)

(43) 公開日 平成17年2月24日(2005.2.24)

(51) Int.CI.⁷
G06F 12/00
G06F 1/00
G06F 12/14

F 1
G06F 12/00
G06F 12/14
G06F 9/06

テーマコード(参考)
5B017
5B076
5B082

審査請求 未請求 請求項の数 6 O.L (全 15 頁)

(21) 出願番号 特願2003-202795 (P2003-202795)
(22) 出願日 平成15年7月29日 (2003.7.29)

特許法第30条第1項適用申請有り 平成15年2月20日 奈良先端科学技術大学院大学主催の「平成14年度博士前期課程修士論文・課題研究発表会」において文書をもって発表

(71) 出願人 504143441
国立大学法人 奈良先端科学技術大学院大学
奈良県生駒市高山町8916-5
(74) 代理人 100067828
弁理士 小谷 悅司
(74) 代理人 100096150
弁理士 伊藤 孝夫
(74) 代理人 100099955
弁理士 桶口 次郎
(74) 代理人 100109438
弁理士 大月 伸介
(72) 発明者 門田 晓人
奈良県生駒市高山町8916-5 奈良先端科学技術大学院大学内

最終頁に続く

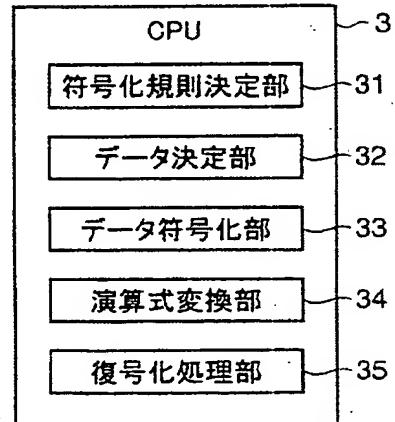
(54) 【発明の名称】プログラム難読化装置、プログラム難読化プログラム及びプログラム難読化方法

(57) 【要約】

【課題】秘密データやアルゴリズムを読み取りにくくするとともに、スタックメモリの覗き見によるプログラムの秘密データの取得も困難にでき、さらには、一般的なプログラムに対して適用することができる難読化手法によりプログラムを難読化する。

【解決手段】プログラム中のデータを符号化規則C($a, b = a \times b$)を用いて符号化するデータ符号化部3と、データ符号化部33によって符号化された符号化データを含む演算式の各演算子に、演算子変換規則R($a, b \sim R10(a, b)$)を適用することにより演算式を符号化演算式に変換する演算式変換部34と、符号化演算式を復号化規則D(a, b)を用いて復号化演算式とする復号化処理部35とを備える。

【選択図】 図2



【特許請求の範囲】

【請求項 1】

データ及びそのデータを演算する演算式を含むプログラムを難読化するプログラム難読化装置であって、

前記データを予め定められた符号化規則に基づいて符号化データに変換する符号化手段と

前記符号化規則に対応する演算子変換規則を用いて前記演算式に含まれる演算子を変換することにより前記演算式を符号化演算式に変換する演算式変換手段と、

前記符号化規則に対応する復号化規則を用いて前記演算式に含まれる演算子を変換することにより前記符号化演算式を復号化演算式に変換する復号化手段とを備え、

10

前記演算子変換規則は、少なくとも加算、減算、乗算及び除算のそれぞれの演算子に対応する演算子変換規則を備え、前記符号化データを前記符号化演算式に代入したときに得られる第1の演算値及び符号化されていないデータを前記演算式に代入したときに得られる第2の演算値の関係が、第1の演算値 = 第2の演算値を前記符号化規則で符号化した値、となるように演算子を変換し、

前記復号化規則は、前記復号化演算式に前記符号化データを代入したときに得られる第3の演算値が前記第2の演算値となるように演算子を変換することを特徴とするプログラム難読化装置。

【請求項 2】

20

前記データは複数種類のデータを含み、

前記データ符号化手段によって符号化されるデータを前記複数種類のデータから決定するデータ決定手段をさらに備え、

前記演算式変換手段は、前記データ決定手段によって決定されたデータを含む演算式を前記演算子変換規則で変換することを特徴とする請求項1記載のプログラム難読化装置。

【請求項 3】

前記符号化規則は、線形変換式 $X = aX + b$ で表され (a, b は定数)、

前記演算子変換規則は、式 (1) ~ (10) で表され、

前記復号化規則は、式 (11) で表されることを特徴とする請求項1又は2記載のプログラム難読化装置。

$$X + Y \rightarrow X + Y - b$$

30

... 式 (1)

$$X - Y \rightarrow X - Y + b$$

... 式 (2)

$$X * Y \rightarrow (X * Y - b * (X + Y - b - a)) / a$$

... 式 (3)

$$X / Y \rightarrow (a * X + b * Y - b * (b + a)) / (Y - b)$$

... 式 (4)

$$X + y \rightarrow X + a * y$$

... 式 (5)

$$X - y \rightarrow X - a * y$$

40

... 式 (6)

$$y - X \rightarrow a * y - X + 2 * b$$

... 式 (7)

$$y * X \rightarrow y * X - b * y + b$$

... 式 (8)

$$X / y \rightarrow (X - b) / y + b$$

... 式 (9)

$$y / X \rightarrow (a * a * y) / (X - b) + b$$

... 式 (10)

$$X \rightarrow (X - b) / a$$

50

式 (11)

但し、 x 及び y は符号化されていないデータを表し、 X 及び Y はそれぞれ x 及び y を前記符号化規則で符号化した符号化データを示す。

【請求項 4】

前記 a 及び b を決定することにより前記符号化規則を決定する符号化規則決定手段をさらに備えることを特徴とする請求項 3 記載のプログラム難読化装置。

【請求項 5】

データ及びそのデータを演算する演算式を含むプログラムを難読化する処理をコンピュータに実行させるプログラム難読化プログラムであって、

前記コンピュータを、

前記データを予め定められた符号化規則に基づいて符号化データに変換する符号化手段と、

前記符号化規則に対応する演算子変換規則を用いて前記演算式に含まれる演算子を変換することにより前記演算式を符号化演算式に変換する演算式変換手段と、

前記符号化規則に対応する復号化規則を用いて前記演算式に含まれる演算子を変換することにより前記符号化演算式を復号化演算式に変換する復号化手段として機能させるとともに、

前記演算子変換規則は、少なくとも加算、減算、乗算及び除算のそれぞれの演算子に対応する演算子変換規則を備え、前記符号化データを前記符号化演算式に代入したときに得られる第1の演算値及び符号化されていないデータを前記演算式に代入したときに得られる第2の演算値の関係が、第1の演算値 = 第2の演算値を前記符号化規則で符号化した値、となるように演算子を変換し、

前記復号化規則は、前記復号化演算式に前記符号化データを代入したときに得られる第3の演算値が前記第2の演算値となるように演算子を変換することを特徴とするプログラム難読化プログラム。

【請求項 6】

データ及びそのデータを演算する演算式を含むプログラムを難読化するプログラム難読化方法であって、

コンピュータが、前記データを予め定められた符号化規則に基づいて符号化データに変換するステップと、

コンピュータが、前記符号化規則に対応する演算子変換規則を用いて前記演算式に含まれる演算子を変換することにより前記演算式を符号化演算式に変換するステップと、

コンピュータが、前記符号化規則に対応する復号化規則を用いて前記演算式に含まれる演算子を変換することにより前記符号化演算式を復号化演算式に変換するステップとを備え、

前記演算子変換規則は、少なくとも加算、減算、乗算及び除算のそれぞれの演算子に対応する演算子変換規則を備え、前記符号化データを前記符号化演算式に代入したときに得られる第1の演算値及び符号化されていないデータを前記演算式に代入したときに得られる第2の演算値の関係が、第1の演算値 = 第2の演算値を前記符号化規則で符号化した値、となるように演算子を変換し、

前記復号化規則は、前記復号化演算式に前記符号化データを代入したときに得られる第3の演算値が前記第2の演算値となるように演算子を変換することを特徴とするプログラム難読化方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、解析が困難なコンピュータプログラムを作成する技術に関する。

【0002】

【従来の技術】

近年、プログラムの解析防止を目的として、プログラムを難読化する方式が多数提案され 50

ている。難読化 (obfuscation) とは、与えられたプログラムを、人間にとつて理解や解析が困難である複雑な（読みにくい）プログラムに変換する技術であり、曖昧化、混乱化とも呼ばれる。

【0003】

プログラムは数多くの構文要素から構成されるため、それぞれの構文要素に対して異なる難読化の方式が存在しうる。そのため、従来提案されている難読化の方式も非常に多岐にわたっている。例えば、プログラム中のシンボリックな情報（変数名や手続き名）を人間にとつて意味のない文字列に置換する方式（特許文献1）、プログラム中の制御構造を複雑にする方式（非特許文献1, 2, 3）、実行結果に影響を与えないダミーコードをプログラム中に挿入する方式（非特許文献4, 5）、手続き（関数）の分割や結合を行う方式（非特許文献6, 7, 8）、プログラム中のデータを符号化する方式（特許文献2, 非特許文献4, 9）、符号化したデータを含む演算式から正規の演算値を得るために演算式の取り扱い方を示したもの（非特許文献10）、プログラム中の複数のデータ（変数）を合成し、別の複数のデータ（変数）として再構成する方式（非特許文献11）等が公表されている。
10

【0004】

さらに、近年、プログラム実行中にスタックメモリに格納されるデータを覗き見することによりプログラムに含まれる秘密データを取得する手法が公表されている（非特許文献1
2）。

【0005】

【特許文献1】

United States Patent, No. 6, 102, 966.

【特許文献2】

特表2002-514333 "ソフトウェアセキュリティを増強するための混乱化技術"
20

【非特許文献1】

C. Collberg and C. Thomborson, "Watermarking, tamper-proofing, and obfuscation - Tools for software protection," IEEE Transactions on Software Engineering, Vol. 28, No. 6, 2002.
30

【非特許文献2】

門田暁人, 高田義広, 鳥居宏次, "プログラムの難読化法の提案", 第51回情報処理学会全国大会, pp. 4-263-4-264, Sep. 1995.

【非特許文献3】

門田暁人, 高田義広, 鳥居宏次, "ループを含むプログラムを難読化する手法の提案," 電子情報通信学会論文誌, Vol. J80-D-I, No. 7, pp. 664-652, 1997.

【非特許文献4】

C. Collberg, C. Thomborson, and D. Low, "A taxonomy of obfuscating transformations," Technical Report of Dept. of Computer Science, University of Auckland, No. 148, New Zealand, 1997.
40

【非特許文献5】

C. Collberg, C. Thomborson, and D. Low, "Manufacturing cheap, resilient, and stealthy opaque constructs," In Proceedings of ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL'98 50

) , San Diego, California, 1998.

【非特許文献6】

小木曾俊夫, 刑部裕介, 双紙正和, 宮地充子, "手続き間呼び出しの関係に着目した難読化手法の提案とその評価," 暗号と情報セキュリティシンポジウム (SCIS 2002), No. 6C-1, 2002.

【非特許文献7】

刑部裕介, 双紙正和, 宮地充子, "オブジェクト指向言語の難読化の提案," 電子情報通信学会技術報告, No. ISEC02-6, 2002.

【非特許文献8】 H. Goto, M. Mambo, H. Shizuya, Y. Watanabe, "Evaluation of tamper-resistant software deviating from structured programming rules," In Proceedings of Australasian Conference on Information Security and Privacy (ACISP2001), Lecture Notes in Computer Science, Vol. 2119, pp. 145-158, 2001.

10

【非特許文献9】

C. Collberg, C. Thomborson, D. Low, "Obfuscation techniques for enhancing software security," International Application Published under the Patent Cooperation Treaty (PCT), International Publication Number: WO 99/01815, International Publication Data: 14 January 1999.

20

【非特許文献10】

T. Sander and C. Tschudin, "Towards Mobile Cryptography", In Proceedings of the IEEE Symposium on Security and Privacy, Oakland, California, May 1998.

30

【非特許文献11】

Fritz Hohl, "An approach to solve the problem of malicious hosts," Technical Report of Fakultaet Informatik, Universitaet Stuttgart, Germany, TR-1997-03, March 1997.

【非特許文献12】

赤井健一郎, 三澤学, 松本勉, "ランタイムデータ探索型耐タンパー性評価法," 情報処理学会論文誌, Vol. 43, No. 8, pp. 2447-2457, 2002.

【0006】

40

【発明が解決しようとする課題】

しかしながら、非特許文献1~3, 5~8, 11及び特許文献1の方式は、いずれもプログラム中のデータを符号化していないため、プログラム中のデータを隠蔽したい場合に役立たない。また、非特許文献12の手法を用いてスタックメモリを覗き見した場合、プログラムに含まれる秘密データが取得される危険性がある。また、データを符号化する方式である特許文献2及び非特許文献4, 10では、制御変数*i*を符号化する簡単な例が示されているだけであり、制御変数*i*以外のデータを符号化することが示されていないため、一般的なプログラムに適用することができないという問題がある。さらに、符号化されたデータを含む演算式の取り扱い方を示した非特許文献10の手法は、特定の演算式(多项式で表される演算)にしか適用することができず、一般的なプログラムに適用することができない。

50

できないという問題がある。

【0007】

本発明の目的は、プログラムを読んだときに秘密データやアルゴリズムを読み取りにくくするとともに、STACKメモリの覗き見によるプログラムの秘密データの取得も困難にでき、さらには、一般的なプログラムに対して適用することができる難読化手法によりプログラムを難読化することができるプログラム難読化装置、プログラム難読化プログラム及びプログラム難読化方法を提供することを目的とする。

【0008】

【課題を解決するための手段】

本発明のプログラム難読化装置は、データ及びそのデータを演算する演算式を含むプログラムを難読化するプログラム難読化装置であって、前記データを予め定められた符号化規則に基づいて符号化データに変換する符号化手段と、前記符号化規則に対応する演算子変換規則を用いて前記演算式に含まれる演算子を変換することにより前記演算式を符号化演算式に変換する演算式変換手段と、前記符号化規則に対応する復号化規則を用いて前記演算式に含まれる演算子を変換することにより前記符号化演算式を復号化演算式に変換する復号化手段とを備え、前記演算子変換規則は、少なくとも加算、減算、乗算及び除算のそれぞれの演算子に対応する演算子変換規則を備え、前記符号化データを前記符号化演算式に代入したときに得られる第1の演算値及び符号化されていないデータを前記演算式に代入したときに得られる第2の演算値の関係が、第1の演算値=第2の演算値を前記符号化規則で符号化した値、となるように演算子を変換し、前記復号化規則は、前記復号化演算式に前記符号化データを代入したときに得られる第3の演算値が前記第2の演算値となるように演算子を変換することを特徴とする。
10

20

【0009】

この構成によれば、プログラム中のデータは、予め定められた符号化規則によって符号化データされ、符号化データを含む演算式は、その符号化規則に対応する演算子変換規則によって各演算子が変換され、符号化演算式に変換される。この演算子変換規則は、第1の演算値=第2の演算値を符号化規則で符号化した値、となるように演算子を変換するものである。

【0010】

したがって、符号化されたデータ及び符号化演算式を含むプログラムを実行した場合、STACKメモリを覗き見しても、元のデータがSTACKメモリに格納されず、また、演算式も元の演算式とは異なる式となっているため、STACKメモリの覗き見によるプログラムに含まれる秘密データの取得を困難にすることができる。さらに、演算子変換規則は、少なくとも加算、減算、乗算及び除算の演算子のそれぞれに対応する演算子変換規則を備えているため、加算、減算、乗算及び除算等の演算子からなる一般的な演算式を含むようなプログラムを難読化することができる。
30

30

【0011】

また、前記データは複数種類のデータを含み、前記演算式は複数種類の演算式を含み、前記データ符号化手段によって符号化されるデータを前記複数のデータから決定するデータ決定手段をさらに備え、前記演算式変換手段は、前記データ決定手段によって決定されたデータを含む演算式を前記演算子変換規則で変換することが好ましい。
40

40

【0012】

この構成によれば、プログラム中に複数種類のデータが存在する場合、そのいずれか、又は全てを符号化することができる。符号化されるデータの種類が多いほど、プログラムは解析が困難となるため、プログラムの解析をより困難化することができる。一方、全てのデータを符号化するのではなく、解読されたくないデータのみを符号化することにより、符号化に費やされる計算量を減少させ、処理の高速化を図ることができる。

【0013】

また、前記符号化規則は $X = a x + b$ で表され (a, b は定数)、前記演算子変換規則は、式 (1) ~ (10) で表され、前記復号化規則は、式 (11) で表されることが好まし
50

い。

【0014】

$$X + Y \rightarrow X + Y - b \\ \dots \text{式 (1)}$$

$$X - Y \rightarrow X - Y + b \\ \dots \text{式 (2)}$$

$$X * Y \rightarrow (X * Y - b * (X + Y - b - a)) / a \\ \dots \text{式 (3)}$$

$$X / Y \rightarrow (a * X + b * Y - b * (b + a)) / (Y - b) \dots \text{式 (4)}$$

$$X + y \rightarrow X + a * y \\ \dots \text{式 (5)}$$

$$X - y \rightarrow X - a * y \\ \dots \text{式 (6)}$$

$$y - X \rightarrow a * y - X + 2 * b \\ \dots \text{式 (7)}$$

$$y * X \rightarrow y * X - b * y + b \\ \dots \text{式 (8)}$$

$$X / y \rightarrow (X - b) / y + b \\ \dots \text{式 (9)}$$

$$y / X \rightarrow (a * a * y) / (X - b) + b \\ \dots \text{式 (10)}$$

$$X \rightarrow (X - b) / a \\ \dots \text{式 (11)}$$

但し、x 及び y は符号化されていないデータを表し、X 及び Y は x 及び y をそれぞれ前記符号化規則で符号化した符号化データを示す。

【0015】

この構成によれば、 $a x + b$ というような線形変換式で表される簡略的な処理でデータを符号化することができる。

【0016】

また、前記 a 及び b を決定することにより前記符号化規則を決定する符号化規則決定手段をさらに備えることが好ましい。この構成によれば、変換式の定数である a 及び b を定めるという簡略化された処理により、符号化規則を決定することができる。ここで、a 及び b の決定は、本装置が決定してもよいし、ユーザが本装置に a 及び b の値を入力することにより決定してもよい。

【0017】

本発明に係るプログラム難読化プログラムは、データ及びそのデータを演算する演算式を含むプログラムを難読化する処理をコンピュータに実行させるプログラム難読化プログラムであって、前記コンピュータを、前記データを予め定められた符号化規則に基づいて符号化データに変換する符号化手段と、前記符号化規則に対応する演算子変換規則を用いて前記演算式に含まれる演算子を変換することにより前記演算式を符号化演算式に変換する演算式変換手段と、前記符号化規則に対応する復号化規則を用いて前記演算式に含まれる演算子を変換することにより前記符号化演算式を復号化演算式に変換する復号化手段として機能させるとともに、前記演算子変換規則は、少なくとも加算、減算、乗算及び除算のそれぞれの演算子に対応する演算子変換規則を備え、前記符号化データを前記符号化演算式に代入したときに得られる第1の演算値及び符号化されていないデータを前記演算式に代入したときに得られる第2の演算値の関係が、第1の演算値 = 第2の演算値を前記符号化規則で符号化した値、となるように演算子を変換し、前記復号化規則は、前記復号化演算式に前記符号化データを代入したときに得られる第3の演算値が前記第2の演算値となるように演算子を変換することを特徴とする。

【0018】

10

20

30

40

50

本プログラム難読化方法は、データ及びそのデータを演算する演算式を含むプログラムを難読化するプログラム難読化方法であって、コンピュータが、前記データを予め定められた符号化規則に基づいて符号化データに変換するステップと、コンピュータが、前記符号化規則に対応する演算子変換規則を用いて前記演算式に含まれる演算子を変換することにより前記演算式を符号化演算式に変換するステップと、コンピュータが、前記符号化規則に対応する復号化規則を用いて前記演算式に含まれる演算子を変換することにより前記符号化演算式を復号化演算式に変換するステップとを備え、前記演算子変換規則は、少なくとも加算、減算、乗算及び除算のそれぞれの演算子に対応する演算子変換規則を備え、前記符号化データを前記符号化演算式に代入したときに得られる第1の演算値及び符号化されていないデータを前記演算式に代入したときに得られる第2の演算値の関係が、第1の演算値=第2の演算値を前記符号化規則で符号化した値、となるように演算子を変換し、前記復号化規則は、前記復号化演算式に前記符号化データを代入したときに得られる第3の演算値が前記第2の演算値となるように演算子を変換することを特徴とする。
10

【0019】

【発明の実施の形態】

以下、本発明の一実施の形態によるプログラム難読化装置について図面を参照しながら説明する。図1は、本発明の一実施の形態によるプログラム難読化装置のハードウェア構成を示すブロック図である。

【0020】

図1に示すプログラム難読化装置は、通常のコンピュータから構成され、入力装置1、ROM(リードオンリーメモリ)2、CPU(中央演算処理装置)3、RAM(ランダムアクセスメモリ)4、外部記憶装置5、表示装置6及び記録媒体駆動装置7を備えている。各ブロックは内部のバスに接続され、このバスを介して種々のデータ等が入出力され、CPU3の制御の下、種々の処理が実行される。
20

【0021】

入力装置1は、キーボード、マウス等から構成され、操作者が種々のデータ及び操作指令等を入力するために使用される。

【0022】

ROM2には、BIOS(Basic Input/Output System)等のシステムプログラム等が記憶されている。外部記憶装置5は、ハードディスクドライブ等から構成され、所定のOS(Operating System)及び後述するプログラム難読化プログラム等が記憶されている。RAM4は、CPU3の作業領域等として用いられる。
30

【0023】

記録媒体駆動装置7は、CD-ROMドライブ、フレキシブルディスクドライブ等から構成される。なお、プログラム難読化プログラムを、CD-ROM、フレキシブルディスク等のコンピュータ読み取り可能な記録媒体8に記録し、記録媒体駆動装置7により記録媒体8からプログラム難読化プログラムを読み出して外部記憶装置5にインストールして実行するようにしてもよい。また、プログラム難読化装置が通信装置等を備え、プログラム難読化プログラムが通信ネットワークを介して接続された他のコンピュータに記憶されている場合、当該コンピュータからネットワークを介してプログラム難読化プログラムをダウンロードして実行するようにしてもよい。
40

【0024】

表示装置6は、液晶表示装置、CRT(陰極線管)等から構成され、CPU3の制御の基、種々の画像を表示する。

【0025】

図2は、CPU3が備える機能を示したブロック図である。CPU3は、符号化規則決定部31、データ決定部32、データ符号化部33、演算式変換部34及び復号化処理部35を機能的に備えており、これらの機能は、CPU3がプログラム難読化プログラムを実行することによって実現される。
50

【0026】

符号化規則決定部31は、プログラムのデータ x を符号化規則により変換するための符号化規則を決定する。本実施形態では、データ x は、 $X = ax + b$ によって線形変換されて X に符号化されるため、符号化規則決定部31は、定数 a 及び b を決定することにより、符号化規則を決定する。ここで、符号化規則決定部31は、定数 a 及び b をランダムに決定してもよいし、オペレータによって入力装置1から入力されたデータを定数 a 及び b としてもよい。なお、 a 、 b はともに整数であり、 $a \geq 1$ である。

【0027】

データ決定部32は、プログラム中の複数種類のデータの中から符号化するデータを決定する。ここで、データ決定部32は、符号化するデータをランダムに決定してもよいし、オペレータによって入力装置1を介して指定されたデータを符号化するデータとして決定してもよい。
10

【0028】

データ符号化部33は、データ決定部32が決定したデータを、符号化規則決定部31が決定した符号化規則を用いて符号化する。本実施形態では、符号化規則は、 $ax + b$ で表されるため、データ符号化部33は、データ決定部32によって決定されたデータ x に、 $X = ax + b$ の演算を施すことでデータ x を符号化データ X に符号化する。

【0029】

演算式変換部34は、符号化されたデータを含む演算式の全ての演算子に対し、以下に示す演算子変換規則1～10を優先順に適用して演算式を符号化演算式に変換する。ここで、優先順とは、一般的な多項式に用いられる演算式の優先順（演算順）と同じ順序である。例えは、式中、左に位置する演算子の方が右に位置する演算子よりも先に演算され、乗算及び除算は加算及び減算よりも先に演算される。
20

変換規則1 $R_1(a, b) : X + Y \rightarrow X + Y - b$

変換規則2 $R_2(a, b) : X - Y \rightarrow X - Y + b$

変換規則3 $R_3(a, b) : X * Y \rightarrow (X * Y - b * (X + Y - b - a)) / a$

変換規則4 $R_4(a, b) : X / Y \rightarrow (a * X + b * Y - b * (b + a)) / (Y - b)$
)

変換規則5 $R_5(a, b) : X + y \rightarrow X + a * y$

変換規則6 $R_6(a, b) : X - y \rightarrow X - a * y$

30

変換規則7 $R_7(a, b) : y - X \rightarrow a * y - X + 2 * b$

変換規則8 $R_8(a, b) : y * X \rightarrow y * X - b * y + b$

変換規則9 $R_9(a, b) : X / y \rightarrow (X - b) / y + b$

変換規則10 $R_{10}(a, b) : y / X \rightarrow (a * a * y) / (X - b) + b$

【0030】

ここに示した符号化規則C(a, b)及び演算子変換規則R(a, b)は、a, bに関して1対1に対応している。つまり、符号化規則C(a, b)のa, bを決定した段階で演算子変換規則R(a, b)も決定されることとなる。したがって、符号化規則C(2, 1)を用いてデータを符号化した後、演算子変換規則R(3, 2)を用いて演算子を変換することはできない。つまり、C(2, 1)を用いてデータを符号化した後は、R(2, 1)を用いる必要がある。
40

【0031】

なお、上記の大文字で示す「X」及び「Y」には符号化されたデータが格納されることを示し、小文字で示す「x」及び「y」は符号化されていないデータが格納されることを示している。例えば、「X + y」は演算子+の両側の値のうち、「X」には符号化されたデータが格納され、「y」には符号化されていないデータが格納されていることを示している。また、変換規則1～10のa及びbの値は、符号化規則決定部31によって決定されたa及びbの値と同じ値が用いられる。これにより、符号化演算式に符号化データXを代入して得られた演算結果（第1の演算値Z1）と、変換前の演算式に符号化されていないデータxを代入したときに得られる演算結果（第2の演算値Z2）とは、 $Z1 = Z2 * a$
50

$+ b$ の関係、すなわち、符号化規則決定部 3 1 で決定された符号化規則の関係を有することとなる。なお、第 2 の演算値 Z_2 は、演算式の正規の演算結果である。

【0032】

復号化処理部 3 5 は、第 1 の演算値 Z_1 が、第 2 の演算値 Z_2 となるような復号化規則を用いて、符号化演算式を復号化演算式に変換する。あるいは、第 1 の演算値 Z_1 を引数として第 2 の演算値 Z_2 を得る命令文をプログラム中に追加する。したがって、復号化演算式に符号化データを代入することにより得られた第 3 の演算値は、第 3 の演算値 = 第 2 の演算値となる。本実施形態では、演算子変換規則が $a \times b$ であるため、復号化規則は以下の関係式で表される。

復号化規則 : $D(a, b) : X \rightarrow (X - b) / a$

10

【0033】

次に、本プログラム難読化装置の処理の概要を簡単な例を用いて説明する。プログラム中のデータ $x = 10$ 、 $y = 20$ と演算 $x * y = 200$ に対して、 x のみを符号化する場合を考える。符号化規則として $C(2, 1)$ を用いると、

$$X = 2 * 10 + 1 = 21$$

により $x = 10$ は $X = 21$ に符号化される。

このとき、 X と y との積 $X * y$ は、変換規則 8 を用いて、次のように変換される。

$$X * y \rightarrow y * X - y + 1 = 20 * 21 - 20 + 1 = 401$$

これは、もとのデータの積 $x * y = 200$ に符号化規則を適用したものとなっている。そして、復号化規則 $D(2, 1)$ 、つまり符号化の逆変換を行えば、元のデータ演算結果が 20 得られることとなる。

$$(401 - 1) / 2 = 200$$

【0034】

図 3 は、本プログラム難読化装置の処理を示したフローチャートである。また、図 4 は、難読化されるプログラムの一例を示した図であり、(a) は難読化前のプログラムを示し、(b) は $p = 1$ に関して難読化されたプログラムを示している。

【0035】

(a) に示すプログラムは、C 言語で記述されたプログラムであり、4 個のデータから 2 個のデータを選ぶ場合の組み合わせの数 (combination) を計算するプログラムである。すなわち、 i を 1 から 2 まで変化させて $p = p * (n - i + 1) / i$ を計算するプログラムである。以下、図 3 のフローチャートに従って、図 4 (a) のプログラムを難読化する処理を説明する。

30

【0036】

まず、ステップ S 1において、符号化規則決定部 3 1 は、 $X = a x + b$ の定数 a 及び b を特定することにより符号化規則 $C(a, b)$ を決定する。以下、 $(a, b) = (2, 1)$ 、すなわち、符号化規則 $C(a, b)$ として $2x + 1$ が決定されたものとする。

【0037】

ステップ S 2において、データ決定部 3 2 は、符号化するデータを決定する。本フローチャートの例では、 $p = 1$ を符号化するデータとして決定している。ここで、符号化規則決定部 3 1 は、符号化するデータとして、 p 以外の n 及び i のうちのいずれか 1 つ、 p 、 n 及び i のうちいずれか 2 つ、又は、 p 、 n 及び i の全てを符号化するデータとして決定してもよい。

40

【0038】

ただし、図 4 の例では、 $p = p * (n - i + 1) / i$ という式を含むため、 n 又は i の符号化が p にも影響を与える。そのため、 p に関する符号化は、必ず行うことが求められる。すなわち、 n 、 i のうちいずれか 1 つのみを、あるいは、 n 、 i の 2 つのみを符号化した場合では、正しい結果は得られない。

【0039】

ステップ S 3において、データ符号化部 3 3 は、ステップ S 1 で決定されたデータを、ステップ S 2 で決定された符号化規則を用いて符号化し符号化データ X とする。この場合、 50

図4 (b) に示すように、 $p = 1$ に $2 * 1 + 1 (= 3)$ の演算が施され、 $p = 1$ が符号化規則 C (2, 1) によって符号化データとされている。これにより、データが隠蔽化される。

【0040】

ステップ S 4において、演算式変換部 34 は、符号化データ X を含む演算式（図4の例では $p = p * (n - i + 1) / i$ ）の演算子に対し、演算規則 1～10 を適用して演算式を符号化演算式 E 2 に変換する。これにより演算式が隠蔽化される。

【0041】

図5は、図4 (a) に示す演算式 E 1 に演算子変換規則が適用される処理を示したフローチャートである。まず、ステップ S 4 1において、アンダーラインで示す $p * (n - i + 1)$ に対し、 p を X、 $n - i + 1$ を y として、変換規則 8 ($y * X \rightarrow y * X - y + 1$) が適用され、 $p * (n - i + 1)$ が $p * (n - i + 1) - (n - i + 1) + 1$ に変換される。
10

【0042】

ステップ S 4 2において、アンダーラインで示す $(p * (n - i + 1) - (n - i + 1) + 1) / i$ に対し、 $p * (n - i + 1) - (n - i + 1) + 1$ を X、 i を y として、変換規則 9 ($X / y \rightarrow (X - 1) / y + 1$) が適用され、 $(p * (n - i + 1) - (n - i + 1) + 1) / i$ が $(p - 1) * (n - i + 1) / i + 1$ に変換される。

【0043】

以上、ステップ S 4 1、S 4 2 の変換処理によって、図4 (a) で示す演算式 E 1 は、図 20 4 (b) で示す符号化演算式 E 2 に変換される。この段階で、符号化演算式 E 2 に $p = 3$ を代入したときに得られる第1の演算値 Z 1 は、演算式 E 1 に $p = 1$ を代入したときの演算値である第2の演算値 Z 2 に対し、 $Z 2 = 2 * Z 1 + 1$ の関係を有している。

【0044】

図3に示すステップ S 5において、復号化処理部 35 は、復号化される必要のある符号化されたデータを復号化する。図4 (a) の例では、プログラムの4行目に演算式 E 1 の演算結果を表示する命令文 I 1 があるため、復号化が必要となる。したがって、この命令文 I 1 のアンダーラインで示す p を X として、復号化規則を適用し、 p を $(p - 1) / 2$ と変更することにより、復号化を行っている。これにより、演算式 E 1 に $p = 1$ を代入したときの演算値である第2の演算値（正規の演算値）を得ることができる。
30

【0045】

ステップ S 6において、全てのデータについて符号化が終了しているか否かが判断され、全てのデータについて符号化が終了している場合（ステップ S 6 で YES）処理が終了され、全てのデータについて処理が終了していない場合（ステップ S 6 で NO）、ステップ S 3 に戻る。図4 (a) の例では、 p のみを符号化しているため、ステップ S 6 で YES と判断され、処理が終了される。

【0046】

次に、本プログラム難読化装置による効果について説明する。図6は、本プログラム難読化装置の効果を説明するための図であり、(a) は難読化される前のプログラムを示し、(b) は難読化されたプログラムを示し、(c) は (a) のトレース結果を示し、(d) は (b) のトレース結果を示している。(a) に示すプログラムは、 i を 1 から 3 まで変化させて、 $p = k * (p * i)$ を計算するプログラムである。このプログラムを符号化規則 C (2, 1) 及び演算子変換規則 R (2, 1) を用いて i , p , k に関して難読化すると、演算式 $p = k * (p * i)$ は、 $p = ((k - 1) * (p * i - p - i + 3) - 2 * k + 6) / 4$ に変換される。また、 $p = 1$ は $p = 3$ に、 $k = 10$ は $k = 21$ に、 $i = 1 \sim 3$ は $i = 3 \sim 7$ に符号化され、また、 $i++ (i = i + 1)$ は演算子変換規則 5 によって $i + = 2 (i = i + 2)$ へ変換される。
40

【0047】

図6 (a) に示すデータ $i (= 1 \sim 3)$ に関してトレースすると、 $p = 10, 200, 6000$ が得られる（(c) 参照）。図6 (b) に示す演算式を $i (= 3 \sim 7)$ に関してト 50

レースすると、 $p = 21, 401, 12001$ が得られる（(d) 参照）。したがって、(b) のプログラムを実行した場合、コンピュータのスタックメモリ上に、符号化される前の p, k, i の値が直接現れないため、スタックメモリを解析しても、元のプログラムに含まれるデータを取得することは困難となる。なお、(d) の $i = 7$ の場合の $p (= 12001)$ を復号化規則 $D(2, 1)$ を用いて復号化すると、 $6000 (= (12001 - 1) / 2)$ となり、正規の値が得られる。

【0048】

なお上記実施形態では、1つのデータ (p) について符号化規則 $C(2, 1)$ を用いて1回だけ符号化しているが、これに限定されず、1つのデータに対して複数回符号化規則 $C(2, 1)$ を適用してデータを変換してもよい。例えば、 $n = 52$ について、符号化規則 $C(2, 1)$ を3回適用すると、 $n = 52$ は1回目の符号化で 105 、2回目の符号化で 211 、3回目の符号化で 423 と符号化される。この場合、 n を含む演算式は、 n が符号化される毎に演算子変換規則 $R(2, 1)$ を用いて n に関して変換してやればよい。10

【0049】

【発明の効果】

請求項1、6、7記載の発明によれば、データを符号化するとともにそのデータを含む演算式を変換するため、プログラムを読んでも秘密データやアルゴリズムを読み取るのが困難になるとともに、スタックメモリに格納されたデータを解析しても、元のプログラムに含まれる秘密データを取得することを困難にすることができる。また、各種演算子に対応する演算子変換規則を用いているため、加算、減算、除算及び乗算等の演算子からなる一般的な演算式を含むプログラムを難読化することができる。20

【0050】

請求項2記載の発明によれば、プログラム中に複数種類のデータが存在する場合、そのいずれか又は複数を符号化することができる。

【0051】

請求項3記載の発明によれば、プログラムの解読をより困難にすることができます。

【0052】

請求項4記載の発明によれば、データを簡略的に符号化することができます。

【0053】

請求項5記載の発明によれば、変換式の定数を定めるだけで、符号化規則を決定することができます。30

【図面の簡単な説明】

【図1】本発明の一実施の形態によるプログラム難読化装置のハードウェア構成を示すブロック図である。

【図2】CPUが備える機能を示したブロック図である。

【図3】本プログラム難読化装置の処理を示したフローチャートである。

【図4】難読化されるプログラムの一例を示した図であり、(a) は難読化前のプログラムを示し、(b) は $p = 1$ に関して難読化されたプログラムを示している。

【図5】図4に示す演算式に演算子変換規則が適用される処理を説明するための図である40

【図6】本プログラム難読化装置の効果を説明するための図であり、(a) は難読化される前のプログラムを示し、(b) は難読化されたプログラムを示し、(c) は (a) のトレース結果を示し、(d) は (b) のトレース結果を示している。

【符号の説明】

1 入力装置

2 ROM

3 CPU

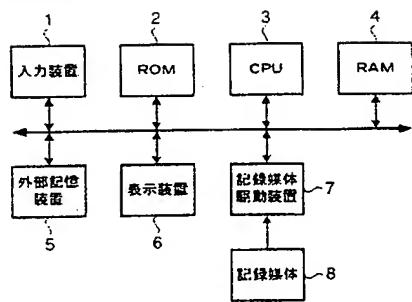
4 RAM

5 外部記憶装置

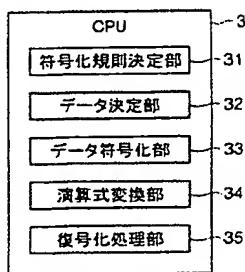
6 表示装置

- 7 記録媒体駆動装置
 8 記録媒体
 3 1 符号化規則決定部
 3 2 データ決定部
 3 3 データ符号化部
 3 4 演算式変換部
 3 5 復号化処理部

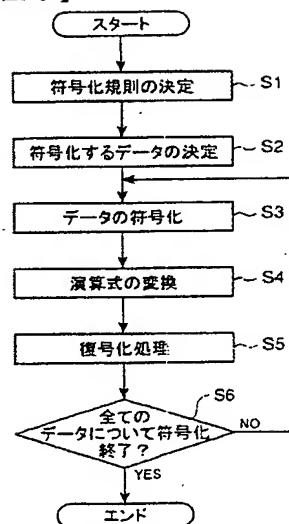
【図 1】



【図 2】



【図 3】



【図 4】

```

int i=1;
int p=1;
int n=4;
for(i=1; i<=2; i++) {
    p=p*(n-i+1)/i;      —— E 1
}
printf("p=%d\n", p);      —— I 1

```

(a)

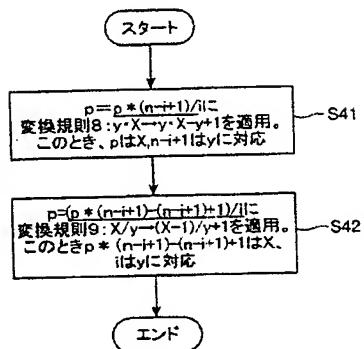
```

int i=1;
int p=3;
int n=4;
for(i=1; i<=2; i++) {
    p=(p-1)*(n-i+1)/i+1;  —— E 2
}
printf("p=%d\n", (p-1)/2);  —— I 2

```

(b)

【図 5】



【図 6】

(a)

```

int i;
int p=1,k=10;
for(i=1; i<=3; i++)
    p=k*(p*i);
p=k*(p*i);

```

(b)

```

int i;
int p=3,k=2;
for(i=3; i<=7; i+=2)
    p=((k-1)*p+i-3)-2*p*k+6)/4;
p=((k-1)*p+((2*i-1)*(2*i-5-21+6))/4-40);

```

(c)

```

(i=3) p=((2*i-1)*(2*i-5-21+6))/4-40
(i=5) p=((2*i-1)*(2*i-5-21+6))/4-40
(i=7) p=((2*i-1)*(2*i-7-3)-2*p*(2*i+6))/4=1200

```

(d)

```

(i=3) p=((2*i-1)*(3*i-3-3+3)-2*p*21+6)/4-21
(i=5) p=((2*i-1)*(X2*i-5-21+3))-2*p*21+6)/4-40
(i=7) p=((2*i-1)*(40*i-7-(01-7-3)-2*p*(2*i+6))/4=1200

```

フロントページの続き

(72)発明者 佐藤 弘紹
千葉市花見川区幕張本郷5-13-10 ノブレス幕張本郷405号

(72)発明者 神崎 雄一郎
奈良県生駒市高山町8916-5 奈良先端科学技術大学院大学内

F ターム(参考) 5B017 AA03 BA07 CA15 CA16
5B076 FA01
5B082 GA02

THIS PAGE BLANK (USPTO)